IN THE SPECIFICATION:

Please amend the paragraph at page 12, lines 14 to 21, as follows:

The active map 115 of the active file system 110 is a bitmap associated with the vacancy of blocks for the active file system 110. The respective snapmaps 140, 145, 150 and 155 are active maps that can be associated with particular snapshots 120, 125, 130 and 135. ~~and an inclusive OR~~ A summary map 160 is an inclusive OR of the snapmaps 140, 145, 150 and 155. Also shown are other blocks 117 ~~115~~ including double indirect blocks 130 and 132, indirect blocks 165, 166 and 167 and data blocks 170, 171, 172 and 173. Finally, Figure 1 shows the spacemap 180 including a collection of spacemap blocks of numbers ~~181,~~ 182, ~~183,~~ 184, 186, 188 and 190.

A1

Please amend the paragraph at page 13, lines 16 to 18, as follows:

In a preferred embodiment, file blocks are 4096 bytes and inodes are 128 bytes. It follows that each block of the inode file contains 32 (i.e. 4,096/128) separate inodes that point to other blocks 117 ~~115~~ in the active file system.

A2

Please amend the paragraphs at page 14, line 6, to page 15, line 7, as follows:

Another inode block in the inode file 105 is inode block N 195 ~~212~~. This block includes a set of pointers to a collection of snapshots 120, 125, 130 and 135 of the volume. Each snapshot includes all the information of a root block and is equivalent to an older root block from a previous active file system. The snapshot 120 may be created at any past CP. Regardless when the snapshot is created, the snapshot is an exact copy of the active file system at that time. The newest snapshot 120 includes a collection of pointers that are aimed directly or indirectly to the same inode file 105 as the root block 100 of the active file system 110.

A3 Cont

As the active file system 110 changes (generally from writing files, deleting files, changing attributes of files, renaming file, modifying their contents and related activities), the active file system and snapshot will diverge over time. Given the slow rate of divergence of an active file system from a snapshot, any two snapshots will share many of the same blocks. The newest snapshot 120 is associated with snapmap 140. Snapmap 140 is a bit map that is initially identical to the active map 115. The older snapshots 125, 130 and 135 ~~194~~ have a corresponding collection of snapmaps 145, 150 and 155. Like the active map 115, these snapmaps 145, 150 and 155 include a set of blocks including bitmaps that correspond to allocated and free blocks for the particular CP when the particular snapmaps 145, 150 and 155 were created. Any active file system may have a structure that includes pointers to one or more snapshots. Snapshots are identical to the active file system when they are created. It follows that snapshots contain pointers to older snapshots. There can be a large number of previous snapshots in any active file

A3 cancel

system or snapshot. In the event that there are no snapshot, there will be no pointers in the active file system.

Please amend the paragraphs at page 16, line 4, to page 17, line 15, as follows:

A summary map 160 is created by using an IOR (inclusive OR) operation 139 on the snapmaps 140, 145, 150 and 155. Like the active map 115 and the snapmaps 140, 145, 150 and 155, the summary map 160 is a file whose data blocks (1, 2, 3, ...Q) contained a bit map. Each bit in each block of the summary map describes the allocation status of one block in the system with "1" being allocated and "0" being free. The summary map 160 describes the allocated and free blocks of the entire volume from all the snapshots 120, 125, 130 and 135 combined. The use of the summary file 160 is to avoid overwriting blocks in use by snapshots.

A4 cont

An IOR operation on sets of blocks (such as 1,024 blocks) of the active map 115 and the summary map 160 produces a spacemap 180. Unlike the active map 115 and the summary map 160, which are a set of blocks containing bitmaps, the spacemap 180 is a set of blocks including ~~181,~~ 182, ~~183,~~ 184, 186, 188 and 190 containing arrays of binary numbers. The binary numbers in the array represent the addition of all the vacant blocks in a region containing a fixed number of blocks, such as 1,024 blocks. The array of binary numbers in the single spacemap block 181 represents the allocation of all blocks for all snapshots and the active file system in one range of 1,024 blocks. Each of the binary numbers ~~181,~~ 182, ~~183,~~ 184, 186, 188

A4 cncld.

and 190 in the array are a fixed length. In a preferred embodiment, the binary numbers are 16 bit numbers, although only 10 bits are used.

In a preferred embodiment, the large spacemap array binary number 182 (0000001111111110=1,021 in decimal units) tells the file system that the corresponding range is relatively full. In such embodiments, the largest binary number 00001111111111 (1,023 in decimal) represents a range containing at most one empty block. The small binary number 184 (0000000000001110=13 in decimal units) instructs the file system that the related range is relatively empty. The spacemap 180 is thus a representation in a very compact form of the allocation of all the blocks in the volume broken into 1,024 block sections. Each 16 bit number in the array of the spacemap 180 corresponds to the allocations of blocks in the range containing 1,024 blocks or about 4 MB. Each spacemap block 180 has about 2,000 binary numbers in the array and they describe the allocation status for 8 GB. Unlike the summary map 120, the spacemap block 180 needs to be determined whenever a file needs to be written.

Please amend the paragraphs at page 18, line 1, to page 19, line 6, as follows:

A5 Cont

The snapmap 205 of the previous active file system, snapshot #1 201, is a bitmap associated with the vacancy of blocks for snapshot #1 201. The respective snapmaps 225, 230 and 235 are earlier active maps that can be associated with particular snapshots 210, 215 and 220. ~~and an inclusive OR~~ A summary map 245 is an inclusive OR of the snapmaps 225, 230 and

235. Also shown are other blocks 211 including double indirect blocks 240 and 241 ~~332~~, indirect blocks 250, 251 and 252, and data blocks 260, 261, 262, and 263 ~~and 264~~. Finally, Figure 2 shows the spacemap 270 of snapshot.#1 201 including a collection of spacemap blocks of binary numbers ~~272, 273, 274, 275 and 276~~.

The old root block 200 includes a collection of pointers that were ~~are~~ written to the previous active file system when the system had reached the previous CP. The pointers are aimed at a set of indirect (or triple indirect, or double indirect) inode blocks (not shown) or directly to the inode file 202 consisting of a set of blocks known as inode blocks 281, 282, 283, 284 and 285.

A5 Cont

An inode block 281 in the inode file 202 points to other blocks 211 ~~328~~ in the old root block 200 ~~201~~ starting with double indirect blocks 240 and 241 ~~332~~ (there could also be triple indirect blocks). The double indirect blocks 240 and 241 ~~332~~ include pointers to indirect blocks 250, 251 and 252. The indirect blocks 250, 251 and 252 include pointers that are directed to data leaf blocks 260, 261, 262, and 263 ~~and 264~~ of the snapshot #1 ~~active file system~~ 201.

Inode block 283 in the inode file 202 points to a set of blocks (1, 2, 3, ..., P) called the snap map 205. Each block in the snap map 205 is a bitmap where each bit corresponds to a block in the entire volume. A "1" in a particular position in the bitmap correlates with a particular allocated block in the snapshot #1 ~~active file system~~ 201. Conversely, a "0" correlates

A5 cmelld.

to the particular block being free for allocation in the old root block 200 ~~201~~. Each block in the snap map 205 can describe up to 32K blocks or 128 MB.

Please amend the paragraph at page 19, lines 13 to 16, as follows:

A6

Snapshot #1 201 also includes an old summary map 245 and old spacemap blocks 270. Although these blocks of data are included in snapshot #1 201 and previous snapshots, in a preferred embodiment, this data is not used by the active file system ~~of figure 2~~.

Please amend the paragraphs at page 20, lines 1 to 16, as follows:

A7 cmit

A method 300 is performed by the file system 110 ~~100~~. Although the method 300 ~~400~~ is described serially, the steps of the method 300 can be performed by separate elements in conjunction or in parallel, whether asynchronously, in a pipelined manner, or otherwise. There is no particular requirement that the method 300 be performed in the same order in which this description lists the steps, except where so indicated.

At a flow point 305, the file system 110 ~~100~~ is ready to perform a method 300.

At a step 310, a user will request a snapshot of the file system 110 ~~100~~.

At a step 315, a timer associated with the file system 110 ~~100~~ initiates the creation of a new snapshot.

At a step 320, the file system 110 ~~100~~ receives a request to make a snapshot.

At a step 325, the file system 110 ~~100~~ creates a new file.

Please amend the paragraphs at page 20, line 21, to page 21, line 5, as follows:

At a step 335, the file system 110 ~~100~~ makes the file read only.

At a step 340, the file system 110 ~~100~~ updates the new summary map by using an inclusive OR of the most recent snapmap and the existing summary file. This step must be done before any blocks are freed in the corresponding active map block. If multiple snapshots are created such that the processing overlaps in time, the update in step 340 need only be done for the most recently created snapshot.

Please amend the paragraphs at page 21, line 14, to page 22, line 22, as follows:

A method 400 is performed by the file system 110 ~~100~~. Although the method 400 is described serially, the steps of the method 400 can be performed by separate elements in

conjunction or in parallel, whether asynchronously, in a pipelined manner, or otherwise. There is no particular requirement that the method 400 be performed in the same order in which this description lists the steps, except where so indicated.

At a flow point 410, the file system 110 ~~100~~ is ready to update the summary map.

At a step 411, update of the summary map is triggered by a "snapdelete" command from an operator or user. As part of this step, the file system 110 ~~100~~ receives and recognizes the "snapdelete" command.

A9
Cmit

At a step 412, the file system 110 ~~100~~ responds immediately to the operator or user, and is ready to receive another operator or user command. However, while the operator or user sees a substantially immediate response, the file system 110 ~~100~~ continues with the method 400 to process the "snapdelete" command.

At a step 413, the file system 110 ~~100~~ marks an entry in the fsinfo block to show that the selected snapshot (designated by the "snapdelete" command) has been deleted.

At a step 414, the file system 110 ~~100~~ examines the snapmap for the selected snapshot for blocks that were in use by the selected snapshot, but might now be eligible to be freed.

A9 Cncld.

At a step 415, the file system 110 ~~100~~ examines the snapmaps for (A) a snapshot just prior to the selected snapshot, and (B) a snapshot just after the selected snapshot. For blocks that were in use by the selected snapshot, the file system 110 ~~100~~ sets the associated bit to indicate the block is FREE, only if both of those snapmaps show that the block was free for those snapshots as well.

Please amend the paragraphs at page 23, lines 4 to 12, as follows:

A10

At a step 421, update of the summary map is triggered by a write allocation operation by the file system 110 ~~100~~. In a preferred embodiment, a write allocation operation occurs for a selected section of the mass storage. The "write allocation" operation refers to selection of free blocks to be seized and written to, as part of flushing data from a set of memory buffers to mass storage. As part of this step, the file system 110 ~~100~~ determines a portion of the summary map corresponding to the selected section of the mass storage.

At a step 422, the file system 110 ~~100~~ recalculates the summary map for the portion of the summary map corresponding to the selected section of the mass storage.

Please amend the paragraphs at page 23, lines 16 to 21, as follows:

At a step 431, update of the summary map is triggered by a background operation. In a preferred embodiment, the file system 110 ~~100~~ updates about one 4K data block of the summary map.

A11

At a step 432, the file system 110 ~~100~~ recalculates the summary map for the portion of the summary map selected to be updated.